

# 基于 HDLC 介质的嵌入式 Linux 二层网桥交换技术<sup>①</sup>

谢天明, 林锡龙

(广州海格通信集团股份有限公司, 广州 515700)

**摘要:** 将 HDLC 设备虚拟成 Linux 下的网口设备, 利用 Linux 下的网桥配置实现 HDLC 协议与以太网之间的互联. 主要目的是介绍虚拟网口驱动程序设计要点和 Linux 网桥搭建技术, 为通信设备不同介质之间的接入实现网桥功能提出了嵌入式 Linux 下的解决方案.

**关键词:** 嵌入式; Linux; 网桥; HDLC; 二层交换

## Second Layer Bridge Switch Technology for Embedded Linux Based on HDLC Media

XIE Tian-Ming, LIN Xi-Long

(Guangzhou Haige Communications Group Co., Ltd, Guangzhou 515700, China)

**Abstract:** This paper makes HDLC device use into a virtual net device in Linux, uses the configuration of bridge in Linux to design for interconnection between HDLC protocol and ethernet. The main purpose of this paper is to introduce the design main points of virtual net device driver and configuration of bridge technology, which provides a scheme for communications equipment between different medias to actualize net bridge function in embedded Linux.

**Key words:** embedded; Linux; bridge; HDLC; second layer switch

网桥是在数据链路层上连接两个网络的设备, 它能够识别数据链路层中的数据帧, 并将这些数据帧临时存储于内存, 再重新生成一个全新的帧转发给相连的另一个网段, 维护 MAC 地址与路径之间的对应关系. 网桥主要完成两个功能: 基于物理网段的 MAC 地址学习并建立 MAC 地址表; 完成数据帧转发. linux 内核支持以太网口的桥接(其他类型的物理接口需要模拟成标准的以太网设备), 可以实现不同物理网段之间的接入, 不需要动态路由协议即可完成选路功能.

本文介绍了一种嵌入式 Linux 系统下创建虚拟以太网设备的方法, 提出实现不同介质之间网桥功能的软件解决方案, 该方案解决了在现有系统不允许升级硬件的约束下实现网络拓扑功能与灵活配置的问题.

## 1 系统设计

系统连接如图 1 所示, 两台运行有 Linux 的嵌入式设备通过 E1 线路连接, 中间使用 HDLC 协议传输数据, PC1 和 PC2 分别连接到设备 A 和设备 B 的网口 1.

在设备 A 和 B 中配置网桥, 网桥下绑定网口 1 和网口 2, 其中网口 2 是基于 E1 接口的虚拟网口, 系统运行后网桥进行 MAC 学习, 建立地址与端口之间的对照表. PC1 和 PC2 之间的数据包可通过设备 A 和 B 进行转发, 从而实现网桥功能.

实际应用中可以将多台嵌入式设备进行拓扑连接, 支持星型、直线型等拓扑结构, 支持多级级联.

## 2 HDLC模拟网口驱动

Linux 系统中对 HDLC 数据包的处理流程如图 2 所示.

Linux 内核中对一个 HDLC 设备是通过数据结构 `hdlc_device` 来表征的, `hdlc_dev` 数据结构中主要实现函数 `xmit`, `xmit` 函数中调用 HDLC 的发送函数将数据发送到物理介质, 读数据则通过启动一个内核线程来不断查询 HDLC 硬件收到的数据包, 并将数据包拼接成网口数据描述结构 `sk_buf`, 最终调用内核协议接口函数 `netif_rx` 交给协议层处理, 到了协议栈以上即可运

<sup>①</sup> 收稿时间:2014-07-14;收到修改稿时间:2016-06-01 [doi:10.15888/j.cnki.csa.005547]

行各种应用如 FTP, HTTP 等. 驱动中最后通过 register\_hdlc\_device 函数将 hdlc\_device 注册到内核进行管理, 如果注册成功, 在系统运行之后采用 ifconfig

命令列举网口设备, 如图 3 所示, 可以看到一个名称为 hdlc 的虚拟以太网设备, 该设备链路层采用 HDLC 传输.

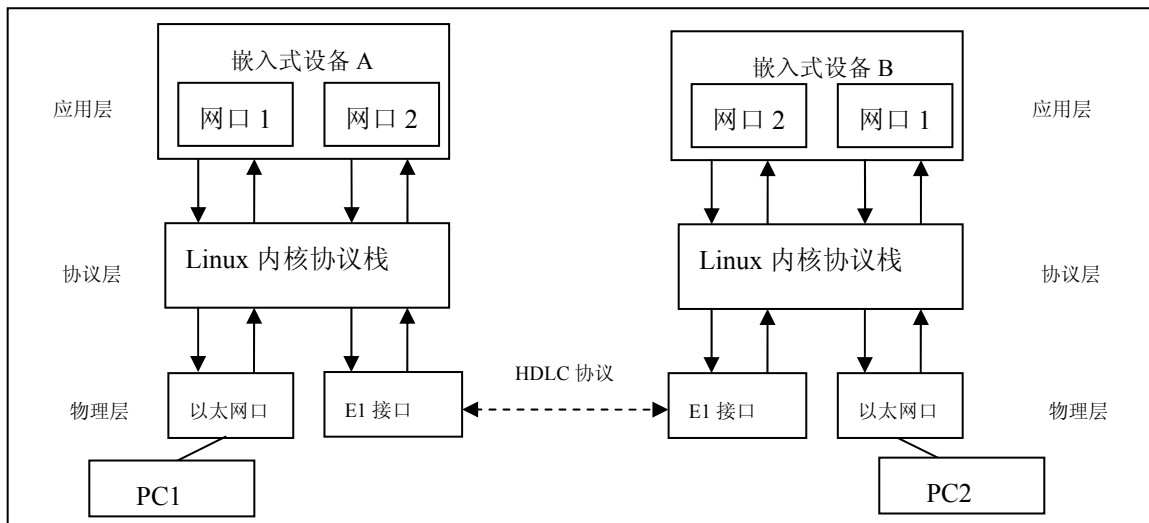


图 1 系统连接示意图

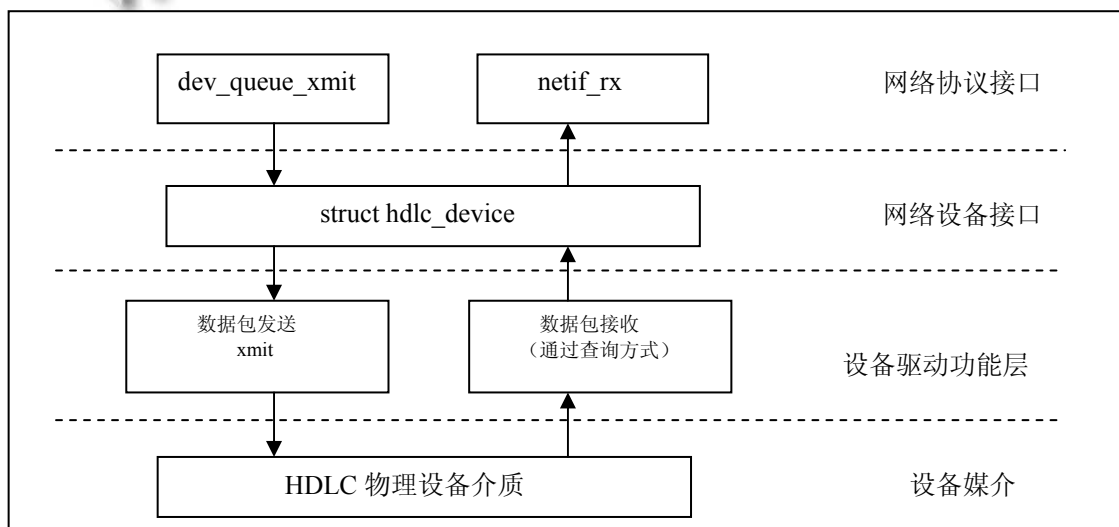


图 2 HDLC 数据包的处理流程

```
# ifconfig hdlc
```

```
hdlc    Link encap:Ethernet HWaddr 00:FF:FF:FF:FF:00
        UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
        RX packets:0 errors:0 dropped:0 overruns:0 frame:0
        TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:500
        RX bytes:0 (0 KiB) TX bytes:0 (0 KiB)
        Memory:96-aa
```

图 3 hdlc 虚拟网口查看设备情况

### 3 Linux网桥技术

linux 内核是通过一个虚拟的网桥设备来实现桥

接的, 该设备可以绑定若干个以太网接口, 图 4 为 Linux 网桥示意图.

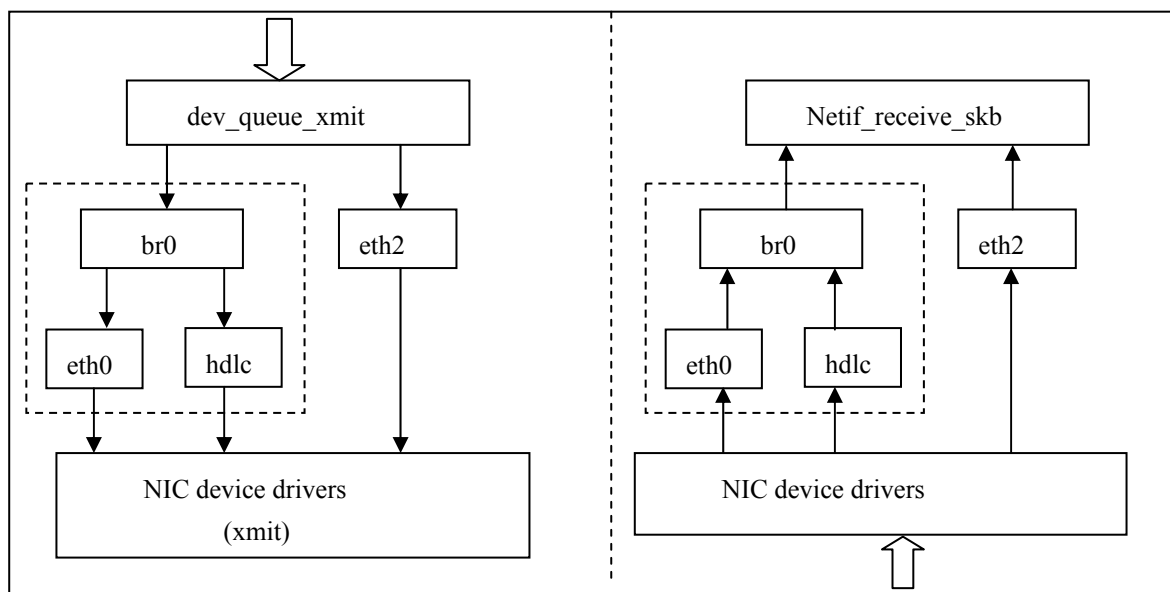


图 4 Linux 网桥示意图

将 `eth0` 和 `hdlc` 两个设备添加到网桥 `br0` 中. 对于网络协议栈的上层来说, 只看得到 `br0`, 因为桥接是在数据链路层实现的, 上层不需要关心桥接的细节. 协议栈上层需要发送的数据被送到 `br0`, 由网桥设备的处理代码再来决定转发到 `eth0` 或是 `hdlc`, 或两者皆是; 反过来, 从 `eth0` 或从 `hdlc` 接收到的数据帧被提交给网桥的处理代码, 决定转发、丢弃、或提交到协议栈上层.

进入桥的数据帧分为几个类型, 桥对应的处理方法也不同:

- ① 报文是发送给自己的, 桥不处理, 交给上层协议栈;
- ② 接收数据帧的物理接口不是网桥接口, 桥不处理, 交给上层协议栈;
- ③ 进入网桥后, 如果网桥的状态为 `Disable`, 则将包丢弃不处理;
- ④ 数据帧源地址无效 (广播, 多播, 以及 `00:00:00:00:00:00`), 丢包;
- ⑤ 如果是 STP 的 BPDU 数据帧, 进入 STP 处理, 处理后不再转发, 也不再交给上层协议栈;
- ⑥ 如果是发给本机的数据帧, 桥直接返回, 交给上层协议栈, 不转发;

需要转发的数据帧分三种情况:

- ① 广播或多播, 则除接收端口外的所有端口都需要转发一份;
- ② 单播并且在 CAM 表中能找到端口映射的, 只需要网映射端口转发一份即可;
- ③ 单播但找不到端口映射的, 则除了接收端口外其余端口都需要转发;

Linux 中使用命令 `brctl` 来配置网桥, 配置顺序为:

```
brctl addbr br0 //虚拟一个网桥 br0
brctl addif br0 hdlc //将 hdlc 网口设备添加到网桥 br0 中
brctl addif br0 eth0 //将 eth0 网口设备添加到网桥 br0 中
ifconfig br0 192.168.1.99 up //启动网桥, 并为其分配 ip 为 192.168.1.99
```

配置完毕之后, 网桥运行后会看到以下信息:

```
br0:port2(eth0) entering learning state
br0:topology change detected,propagating
br0: port2(eth0) entering forwarding state
```

网桥已经进入学习和转发状态, 在图 1 中 PC1 可以 ping 通 PC2, 实现网桥功能. 图 5 显示查看系统建立网桥后网口信息状态.

```

# ifconfig
br0      Link encap:Ethernet  HWaddr 00:75:C0:A8:64:6E
         inet addr:192.168.100.110  Bcast:192.168.100.255  Mask:255.255.255.0
         UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
         RX packets:636 errors:0 dropped:0 overruns:0 frame:0
         TX packets:629 errors:0 dropped:0 overruns:0 carrier:0
         collisions:0 txqueuelen:0
         RX bytes:37810 (36.9 KiB)  TX bytes:45970 (44.8 KiB)

eth0     Link encap:Ethernet  HWaddr 00:E0:C0:48:01:03
         inet addr:192.168.1.3  Bcast:192.168.1.255  Mask:255.255.255.0
         UP BROADCAST MULTICAST  MTU:1500  Metric:1
         RX packets:0 errors:0 dropped:0 overruns:0 frame:0
         TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
         collisions:0 txqueuelen:1000
         RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
         Base address:0x2200

hdlc     Link encap:Ethernet  HWaddr 00:FF:FF:FF:FF:00
         UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
         RX packets:3100 errors:0 dropped:0 overruns:0 frame:0
         TX packets:3111 errors:0 dropped:0 overruns:0 carrier:0
         collisions:0 txqueuelen:500
         RX bytes:228168 (222.8 KiB)  TX bytes:228940 (223.5 KiB)
         Memory:96-aa

```

图 5 系统建立网桥后网口信息状态

可以看出, 在系统中有三个网口设备, 其中, br0 为网桥设备, eth0 为以太网设备, hdlc 为虚拟网口设备。

#### 4 结语

路由控制是指在网络中按照正确的方向转发数据包到达目标主机, 在网络拓扑比较简单的情况下, 特别是在基于嵌入式系统的网络拓扑中, 有时并不需要三层交换功能, 本文提出的基于 Linux 内核自带的网桥功能, 没有涉及到复杂的动态路由协议, 这种方案是采用软件的方式实现的, 相对采用硬件方案实现的网桥技术具有实现简单, 应用灵活, 可扩展性强等优点。另外, 文章中基于 HDLC 的网口驱动设计对于其他物理接口也有很好的借鉴意义。该方案在某通信设

备上得到实现, 运行良好, 平均无故障时间达到预期要求, 这充分验证了设计和实现的正确性。

#### 参考文献

- 1 竹下隆史.图解 TCP/IP.北京:人民邮电出版社,2013.
- 2 Doyle J.TCP/IP 路由技术.北京:人民邮电出版社,2009.
- 3 宋宝华.Linux 设备驱动开发详解.北京:人民邮电出版社, 2010.
- 4 Stevens WR.TCP/IP 详解卷 1: 协议.北京:机械工业出版社, 2009.
- 5 王达.深入理解计算机网络.北京:机械工业出版社,2013.
- 6 三轮贤一.图解网络硬件.北京:人民邮电出版社,2014.